SIAM J. NUMER. ANAL.
Vol. 30, No. 4, pp. 1099–1120, August 1993

010

STABILIZATION OF UNSTABLE PROCEDURES: THE RECURSIVE PROJECTION METHOD*

GAUTAM M. SHROFF† AND HERBERT B. KELLER‡

Abstract. Fixed-point iterative procedures for solving nonlinear parameter dependent problems can converge for some interval of parameter values and diverge as the parameter changes. The Recursive Projection Method (RPM), which stabilizes such procedures by computing a projection onto the unstable subspace is presented. On this subspace a Newton or special Newton iteration is performed, and the fixed-point iteration is used on the complement. As continuation in the parameter proceeds, the projection is efficiently updated, possibly increasing or decreasing the dimension of the unstable subspace. The method is extremely effective when the dimension of the unstable subspace is small compared to the dimension of the system. Convergence proofs are given and pseudo-arclength continuation on the unstable subspace is introduced to allow continuation past folds. Examples are presented for an important application of the RPM in which a "black-box" time integration scheme is stabilized, enabling it to compute *unstable* steady states. The RPM can also be used to accelerate iterative procedures when slow convergence is due to a few slowly decaying modes.

Key words. stabilization procedures, projection methods, stabilized continuation

AMS subject classifications. 65B99, 65M10, 65N20

1. Introduction.

To approximate the solution of many nonlinear parameter dependent problems we are often led to recursive "fixed-point" procedures of the form

$$u^{(\nu+1)} = F(u^{(\nu)}, \lambda). \tag{1.1}$$

Here, $F : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N$ is smooth and $N \gg 1$ (since we are mainly concerned with the discretization of some nonlinear functional equations on a fine grid). If this procedure converges, say $\{u^{(\nu)}(\lambda)\} \to u^*(\lambda)$, then

$$u^*(\lambda) = F(u^*(\lambda), \lambda) \tag{1.2}$$

and the solution may exist on some interval $\lambda_a < \lambda < \lambda_b$. However, the iterative procedure (1.1) need not converge over the entire interval of existence. Indeed, as λ varies, the convergence rate may change, and the procedure may become unstable and diverge. What can be done to recover convergence? In this paper, we consider an important class of such problems and demonstrate an efficient stabilization procedure to correct the situation.

Recall that the iteration (1.1) converges locally in a neighborhood of a solution as long as all the eigenvalues of the Jacobian matrix $F_u^* \equiv F_u(u^*(\lambda), \lambda)$ lie within the unit disk $\{|z| < 1\}$. Our main concern here will be with problems in which the convergence of (1.1) fails along the solution path $\Gamma = \{u^*(\lambda), \lambda\}$ as a few eigenvalues of F_u^* leave the unit disk. We first give some important examples of problems in which this occurs.

Steady states of nonlinear dynamical systems of the form:

$$u_t = G(u, \lambda), \tag{1.3}$$

are also equilibrium solutions of

$$G(u^*(\lambda), \lambda) = 0. \tag{1.4}$$

*Received by the editors June 3, 1991; accepted for publication (in revised form) September 11, 1992. This work was supported in part by the National Science Foundation Cooperative Agreement CCR-9120008 and the Department of Energy Project Agreement No. DE-FG03-89ER25073. The government has certain rights on this material.

†Department of Computer Science and Engineering, Indian Institute of Technology, Delhi 110016, India.
‡Applied Mathematics 217-50, California Institute of Technology, Pasadena California 91125.

Here, $G(u, \lambda)$ is a partial differential operator, $G : \mathbb{B} \times \mathbb{R} \to \mathbb{B}$, and solutions of (1.4) represent steady-state solutions of the underlying physical problem. We often have available a code based on a discretization of the operator G on a suitable grid and a numerical integration of the resulting system of ordinary differential equations. Steady-state solutions are often computed simply by using the available code to integrate the time dependent system (1.3) for large times. We can view the time integration code as a fixed point procedure of the form (1.1), with ν counting the time steps. Such a procedure will fail to converge if the equilibrium states to be computed become unstable during continuation in λ (see Appendix A for details). This usually occurs when only one or two eigenvalues of the linearized stability problem for (1.3) have their real parts change sign. Our stabilization procedure allows us to compute such unstable steady states and entire unstable solution branches via continuation, using this time integration code as a "black box." Such an approach has not been proposed before and should be of great practical significance.

Alternatively, we consider directly computing solutions of the equilibrium equations (1.4). Frequently employed iterative procedures are of the form:

$$u^{(\nu+1)}(\lambda) = u^{(\nu)}(\lambda) - M^{(\nu)}(\lambda)G(u^{(\nu)}(\lambda), \lambda), \qquad \nu = 0, 1, \ldots. \tag{1.5}$$

Here, we assume that the operator G in (1.4) has been approximated by some appropriate discretization, so $G : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N$. The $M^{(\nu)}(\lambda)$ are $N \times N$ matrices, chosen to produce a convergent scheme in an efficient manner. For example $M^{(\nu)}(\lambda) \equiv G_u^{-1}(u^{(\nu)}(\lambda), \lambda)$ yields Newton's method. Many other choices are possible, such as a quasi-Newton method, where $M^{(\nu)}(\lambda)$ is the inverse of some approximate Jacobian. Alternatively, the iteration (1.5) may represent a relaxation procedure such as Gauss–Seidel, SOR, or multigrid. Whatever the choice made for $M^{(\nu)}(\lambda)$ the relation to (1.1) is simply that

$$F(u^{(\nu)}, \lambda) \equiv u^{(\nu)} - M^{(\nu)}(\lambda)G(u^{(\nu)}(\lambda), \lambda). \tag{1.6}$$

This may converge well for some λ, but if during continuation $u^*(\lambda)$ approaches a solution at which $G_u(u^*(\lambda), \lambda)$ is singular then some eigenvalues of $F_u(u^*(\lambda), \lambda)$ approach unity. The generic singular case is when G_u has a one-dimensional null space from which it follows that $F_u(u^*(\lambda), \lambda)$ has one eigenvalue unity. Continuation past such points is usually possible using standard pseudo-arclength techniques [6]. However, we stress that once again the difficulties encountered are due to a small number of eigenvalues leaving the unit disk (for further details, see §7). Our stabilization procedure can dramatically accelerate convergence of such iterations near singular points where the convergence rate normally degrades. It can also extend the domain of convergence beyond the normal range.

Our stabilization procedure: the *Recursive Projection Method* or RPM exploits the fact that, as in the above examples, slow convergence or even divergence of the fixed-point iteration (1.1) is caused by a small number of eigenvalues leaving (or approaching) the unit disk. The key idea is to find or approximate the eigenspace corresponding to the unstable modes. This is done efficiently in a recursive manner, using the iterates of the fixed-point iteration. The space $\mathbb{R}^n$ is written as a direct sum of the span of the unstable eigenspace (say, $\mathbb{P}$) and its orthogonal complement (say, $\mathbb{Q}$). We modify the iteration by performing Newton's method on the subspace $\mathbb{P}$ while continuing to use the iteration (1.1) on $\mathbb{Q}$, where it does converge. The extra work involved will be small as long as the number of eigenvalues of F_u^* near or outside the unit circle remains small.

The basic idea of using a coupled iteration to force convergence of a fixed-point iteration was used by Jarausch and Mackens for solving nonlinear equations [5], as well as in the context of continuation [3], [4]. However, they treat only systems in which F_u^* is symmetric. In this paper, we treat general fixed-point iterations, which include nonsymmetric F_u^*. This requires more care in the computation of the unstable eigenvectors. Further and more important, we show how these eigenvectors can be approximated and maintained directly from the iterates of (1.1) as continuation proceeds; we do not compute Jacobians and perform only minimal additional work in the stabilization procedure.

This paper is organized as follows. The stabilization procedure is developed in §2 followed by a local convergence analysis in §3. In §4, we show how to recursively estimate and maintain a projection onto the unstable eigenspace directly from the iteration itself. In §5, we describe the RPM algorithm, which numerically implements the stabilization procedure using the projections computed by the recursive technique of §4. In §6 we modify the convergence analysis of §3 after considering the effects of some of the approximations made in the numerical implementation. Incorporating pseudo-arclength methods in our algorithm to treat the case of singular F_u^* is considered and briefly analyzed in §7. In §8, we present numerical experiments in which the stabilization algorithm uses a "black box" time integration code to compute steady-state solution branches, which become unstable. Section 9 contains our conclusions and mentions some directions in which the algorithm might be further generalized. Appendix A contains a more detailed examination of the application of our methods to stabilizing unstable time integration schemes.

2. The stabilization procedure. We assume that the problem we are attempting to solve,

$$u = F(u, \lambda), \qquad F : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N, \tag{2.1}$$

has a smooth arc of solutions $\Gamma : u = u^*(\lambda)$ parameterized by $\lambda \in [\lambda_a, \lambda_b]$. To find or approximate this branch of solutions, a predictor-solver continuation procedure is used. Given two points on Γ, say $u_{-1}^* \equiv u^*(\lambda_{-1})$ and $u_0^* \equiv u^*(\lambda_0)$ with $\lambda_{-1} < \lambda_0$ both in $[\lambda_a, \lambda_b]$, we predict a new solution by the secant method:

$$\begin{aligned} &\text{(a)} \quad \lambda \;\; = \lambda_0 + \delta\lambda, \\ &\text{(b)} \quad u^{(0)}(\lambda) = u_0^* + \frac{\delta\lambda}{\lambda_0 - \lambda_{-1}}(u_0^* - u_{-1}^*). \end{aligned} \tag{2.2}$$

Here, $\delta\lambda$ is the step size and the solver is the fixed-point scheme

$$u^{(\nu+1)} = F(u^{(\nu)}, \lambda). \tag{2.3}$$

As we show in §3, such an iteration converges if the eigenvalues $\{\mu_k\}_1^N$ of the Jacobian $F_u^* \equiv F_u(u^*(\lambda), \lambda)$ lie in the unit disk and the initial iterate $u^{(0)}(\lambda)$ is sufficiently close to $u^*(\lambda)$.

However, the scheme generally fails if any of these eigenvalues lie outside the unit disk; even if all eigenvalues are within the unit disk, the convergence may be slow if any of them lie close to the boundary of the disk. Suppose a small number, m, of the eigenvalues lie outside the disk

$$K_\delta = \{|z| \le 1 - \delta\} \quad \text{for some } \delta > 0, \tag{2.4}$$

that is:

$$|\mu_1| \ge \cdots \ge |\mu_m| > 1 - \delta \ge |\mu_{m+1}| \ge \cdots \ge |\mu_N|. \tag{2.5}$$

Define subspaces $\mathbb{P}$ and $\mathbb{Q}$ of $\mathbb{R}^N$ by:

$$\text{(2.6)} \quad \begin{aligned} &\text{(a)} \quad \mathbb{P} \equiv \text{the maximal invariant subspace of } F_u^* \text{ belonging to } \{\mu_k\}_1^m, \\ &\text{(b)} \quad \mathbb{Q} \equiv \mathbb{R}^N - \mathbb{P}, \text{ the orthogonal complement of } \mathbb{P}. \end{aligned}$$

Note that the subspace $\mathbb{Q}$ is *not* in general an invariant subspace of F_u^*. (If F_u^* were normal, then $\mathbb{Q}$ would also be an invariant subspace.) Denote the orthogonal projectors of $\mathbb{R}^N$ onto these subspaces by P and Q, respectively. This induces an orthogonal direct sum decomposition of the space $\mathbb{R}^N$:

$$\text{(2.7)} \qquad \mathbb{R}^N = \mathbb{P} \oplus \mathbb{Q} = P\mathbb{R}^N \oplus Q\mathbb{R}^N,$$

Here, $Q = I - P$ so that $PQ = 0$ (since $P^2 = P$). Using (2.7) we have for each $u \in \mathbb{R}^N$ the unique decomposition:

$$\text{(2.8)} \qquad u = p + q, \quad p \equiv Pu \in \mathbb{P}, \quad q \equiv Qu \in \mathbb{Q}.$$

Now a Lyapunov–Schmidt decomposition of the system $u = F(u, \lambda)$ can be introduced by applying P and Q to (2.1) and using (2.7) to get:

$$\text{(2.9)} \quad \begin{aligned} &\text{(a)} \quad p = f(p, q, \lambda) \equiv PF(p + q, \lambda), \\ &\text{(b)} \quad q = g(p, q, \lambda) \equiv QF(p + q, \lambda). \end{aligned}$$

Clearly, each solution branch $u^*(\lambda)$ of $u = F(u, \lambda)$ corresponds to a solution branch $(p^*, q^*) \equiv (p^*(\lambda), q^*(\lambda))$ of (2.9). We claim that $g(p, q, \lambda)$ in (2.9b) is contracting in q for (p, q) near (p^*, q^*). This is the motivation for our stabilization algorithm, and it follows from the following.

LEMMA 2.10. *All the eigenvalues of*

$$g_q^* = g_q(p^*, q^*, \lambda) = QF_u^*Q$$

lie in the disk K_δ.

The proof will be given in §3.

From this lemma we see that the recursion

$$\text{(2.11)} \qquad q^{(\nu+1)} = g(p, q^{(\nu)}, \lambda)$$

will be locally convergent on $\mathbb{Q}$ in some neighborhood of (p^*, q^*), even though the general iteration (2.3) may not converge on $\mathbb{R}^N$. The idea of our stabilization procedure is to use Newton's method for p on the system $p = f(p, q, \lambda)$ while continuing to use fixed-point iteration for q. This leads to the new scheme:

$$\text{(2.12)} \quad \begin{aligned} &\text{(a)} \quad (I - f_p^{(\nu)})(p^{(\nu+1)} - p^{(\nu)}) = f(p^{(\nu)}, q^{(\nu)}, \lambda) - p^{(\nu)}, \\ &\text{(b)} \quad\qquad\qquad\qquad\quad q^{(\nu+1)} = g(p^{(\nu)}, q^{(\nu)}, \lambda). \end{aligned}$$

Here, we use $u^{(\nu)} = p^{(\nu)} + q^{(\nu)}$ (not the $u^{(\nu)}$ in (2.3)) to define

$$\text{(2.13)} \qquad f_p^{(\nu)} \equiv f_p(p^{(\nu)}, q^{(\nu)}, \lambda) \equiv PF_u(u^{(\nu)}, \lambda)P.$$

Clearly, $f_p^{(\nu)}$ is the restriction of the matrix $F_u(u^{(\nu)}, \lambda)$ to the subspace $\mathbb{P}$. Thus, $(I - f_p^{(\nu)}) : \mathbb{P} \to \mathbb{P}$ has an inverse if the spectrum of $f_p^{(\nu)}$ does not contain unity. If

this fails to hold during our continuation in λ we introduce pseudo-arclength as done in §7. For the present, we assume that $(I - f_p^{(\nu)})^{-1}$ exists on $\mathbb{P} \to \mathbb{P}$. Then our stabilized iteration scheme can be described as follows.

Stabilized Iteration.

$$
\begin{aligned}
&1. \quad p^{(0)} = Pu^{(0)}(\lambda), \quad q^{(0)} = Qu^{(0)}(\lambda);\\
&2. \quad \text{Do until convergence :}\\
&\quad \text{(a) } p^{(\nu+1)} = +p^{(\nu)} + (I - f_p^{(\nu)})^{-1}(f(p^{(\nu)}, q^{(\nu)}, \lambda) - p^{(\nu)}),\\
&\quad \text{(b) } q^{(\nu+1)} = +g(p^{(\nu)}, q^{(\nu)}, \lambda);\\
&3. \quad u^*(\lambda) = p^{(\nu_{\text{Final}})} + q^{(\nu_{\text{Final}})} \equiv p^* + q^*.
\end{aligned} \tag{2.14}
$$

3. Convergence analysis. We proceed to show that (2.14) will converge for $u^{(0)}(\lambda)$ in a neighborhood of the solution $u^*(\lambda)$, even when the original fixed-point iteration (2.3) diverges.

LEMMA 3.1. *The orthogonal projections P and Q onto the subspaces $\mathbb{P}$ and $\mathbb{Q}$ defined in* (2.6) *satisfy*:

$$QF_u^*P = 0.$$

Proof. Since $\mathbb{P}$ is an invariant subspace of F_u^*,

$$F_u^*Pv \in \mathbb{P} \quad \text{for all } v \in \mathbb{R}^N.$$

Therefore, $F_u^*Pv = PF_u^*Pv$, and the result follows from $QP = 0$. □

In the previous section, Lemma 2.10 was used to motivate the stabilization algorithm. We now prove that result.

Proof of Lemma 2.10. We are required to show that the eigenvalues of $g_q^* = QF_u^*Q$ all lie within the disk K_δ. There exists a real block diagonal decomposition [2] of F_u^*,

$$F_u^* = WJW^{-1}, \tag{3.2}$$

where we partition the similarity matrix as

$$W = (W_1, W_2), \quad W_1 \in \mathbb{R}^{N\times m}, \quad W_2 \in \mathbb{R}^{N\times(N-m)}, \tag{3.3}$$

and

$$J = \begin{pmatrix} J_1 & 0 \\ 0 & J_2 \end{pmatrix}, \quad J_1 \in \mathbb{R}^{m\times m}, \quad J_2 \in \mathbb{R}^{(N-m)\times(N-m)}. \tag{3.4}$$

The block J_1 contains all the blocks associated with the eigenvalues $\mu_1, \ldots, \mu_m$, and J_2 contains the blocks for $\mu_{m+1}, \ldots, \mu_N$. Also, the columns of W_1 and W_2 form bases for the invariant subspaces associated with eigenvalues $\mu_1, \ldots, \mu_m$ and $\mu_{m+1}, \ldots, \mu_N$, respectively. So by the definition of $\mathbb{P}$ it follows that $\mathbb{P} = \mathcal{R}(W_1)$, the range space of W_1. Note also that this implies $QW_1 = 0$. Now $F_u^*W_2 = W_2J_2$ from (3.2), so

$$QW_2J_2 = QF_u^*W_2 = QF_u^*(PW_2 + QW_2) = QF_u^*QW_2. \tag{3.5}$$

Here we have used $P + Q = I$ and Lemma 3.1. Let $V = [W_1, QW_2]$, and use (3.5), $Q^2 = Q$ and $QW_1 = 0$, to get

$$QF_u^*QV = QF_u^*Q([W_1, QW_2]) = (W_1, QW_2)\begin{pmatrix} 0 & 0 \\ 0 & J_2 \end{pmatrix}. \tag{3.6}$$

Suppose there exists $w \in \mathbb{R}^{(N-m)}$, $w \neq 0$, such that $QW_2w = 0$. Now $W_2w \neq 0$ since W is nonsingular, so we must have $W_2w \in \mathbb{P} = \mathcal{R}(W_1)$. But this too is impossible since W is nonsingular. Thus, $\text{rank}(QW_2) = N - m$. It follows that $V = [W_1, QW_2]$ is nonsingular, and (3.6) yields

$$QF_u^*Q = V \begin{pmatrix} 0 & 0 \\ 0 & J_2 \end{pmatrix} V^{-1}. \tag{3.7}$$

Since the eigenvalues of J_2 are $\mu_{m+1}, \ldots, \mu_N$, which all lie within K_δ, the result follows. □

Notice that in our stabilization algorithm we have not assumed any special properties of F_u^*, such as symmetry. In the special case of symmetric (or even normal) F_u^* the results of Lemma 2.10 hold for more general $\mathbb{P}$, i.e., we require only that the eigenvectors corresponding to $\mu_1, \ldots, \mu_m$ be *contained* in $\mathbb{P}$. However, in the general nonsymmetric case, the assumption that $\mathbb{P}$ is in fact an invariant subspace of F_u^* is crucial. This is illustrated by the following simple example:

$$F_u^* = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 2 \\ 0 & -2 & -2 \end{pmatrix},$$

whose eigenvalues are $(2, 0, 0)$, and e_1 is the eigenvector corresponding to the eigenvalue $\mu = 2$. However, if we take $\mathbb{P} = \text{span}(e_1, e_2)$, then QF_u^*Q has the eigenvalue $\mu = -2$, which is still outside the unit disk. Because of this phenomenon, our stabilization algorithm will have to exercise greater care in the computation of the projector P than is the case in special algorithms, which assume that F_u^* is symmetric [3], [5].

To complete the convergence analysis of the stabilized iteration (2.14) we write it as

$$\begin{aligned} &\text{(a)} \quad p^{(\nu+1)} = h(p^{(\nu)}, q^{(\nu)}, \lambda), \\ &\text{(b)} \quad q^{(\nu+1)} = g(p^{(\nu)}, q^{(\nu)}, \lambda), \end{aligned} \tag{3.8}$$

where we have introduced

$$h(p, q, \lambda) \equiv p + (I - f_p(p, q, \lambda))^{-1}(f(p, q, \lambda) - p). \tag{3.9}$$

Of course, we can use (3.9) only when $(I - f_p(p, q, \lambda))$ is nonsingular and we assume this to hold at the solution $(p^*(\lambda), q^*(\lambda))$. Then with smoothness of $f(p, q, \lambda)$ it also holds in some neighborhood of $(p^*(\lambda), q^*(\lambda))$. If $f_p(p, q, \lambda)$ is differentiable, we can evaluate the Jacobian $\partial(h, g)/\partial(p, q)$ at the solution to get, recalling that $f(p^*, q^*, \lambda) = p^*$:

$$\begin{aligned} \left.\frac{\partial(h, g)}{\partial(p, q)}\right|_{p^*, q^*} &= \begin{pmatrix} I + (I - f_p^*)^{-1}(f_p^* - I) & (I - f_p^*)^{-1} f_q^* \\ g_p^* & g_q^* \end{pmatrix} \\ &= \begin{pmatrix} 0 & (I - f_p^*)^{-1} f_q^* \\ 0 & g_q^* \end{pmatrix}. \end{aligned} \tag{3.10}$$

Here, we have used $g_p^* = QF_u^*P = 0$ by Lemma 3.1. But Lemma 2.10 ensures that the spectrum of g_q^* is in the disk K_δ. Thus, we have proven the following:

LEMMA 3.11. *The Jacobian of the stabilized iteration* (2.14), *has a spectral radius satisfying*

$$\rho\left(\left.\frac{\partial(h, g)}{\partial(p, q)}\right|_{p^*, q^*}\right) \leq 1 - \delta < 1. \tag{3.12}$$

Using this result we demonstrate the convergence of the stabilized iteration as stated in the following theorem.

THEOREM 3.13. *For some* $\varepsilon_1 = \varepsilon_1(\lambda) > 0$ *let* $F(u,\lambda)$ *satisfy on the ball* $B_{\varepsilon_1}(u^*) = \{u : \|u - u^*\| < \varepsilon_1\}$:

(a) $F(u,\lambda)$ *is twice differentiable*,

(b) $1 \notin \sigma\{f_p^*\} \equiv \sigma\{PF_u(u^*(\lambda),\lambda)P\}$.

Then the stabilized iteration (2.14) *converges for all initial values* $u^{(0)} \in B_\varepsilon(u^*)$ *for some* $\varepsilon < \varepsilon_1$.

Proof. Denote by $v \in \mathbb{R}^{2N}$ the iterates of (3.8) in the form:

$$v^{(\nu)} \equiv \begin{pmatrix} p^{(\nu)} \\ q^{(\nu)} \end{pmatrix} \equiv \begin{pmatrix} Pu^{(\nu)} \\ Qu^{(\nu)} \end{pmatrix}. \tag{3.14}$$

Conditions (a) and (b) ensure that $v^{(\nu+1)}$ is well defined if $p^{(\nu)} + q^{(\nu)} = u^{(\nu)} \in B_{\varepsilon_1}(u^*)$. The initial values satisfy the condition for $\nu = 0$. For an induction, we consider, with

$$v^* \equiv \begin{pmatrix} p^* \\ q^* \end{pmatrix} : v^{(\nu+1)} - v^* = \begin{pmatrix} h(p^{(\nu)}, q^{(\nu)}, \lambda) - h(p^*, q^*, \lambda) \\ g(p^{(\nu)}, q^{(\nu)}, \lambda) - g(p^*, q^*, \lambda) \end{pmatrix},$$

and a Taylor expansion about v^* yields

$$v^{(\nu+1)} - v^* = \left.\frac{\partial(h,g)}{\partial(p,q)}\right|_{p^*,q^*} (v^{(\nu)} - v^*) + O(\|v^{(\nu)} - v^*\|^2).$$

For any square matrix A and any $\eta > 0$ there exists a norm $\|.\|_{A,\eta}$ such that $\|A\|_{A,\eta} < \rho(A) + \eta$. Using such a norm, we get from the above and Lemma 3.11 that:

$$\begin{aligned} \|v^{(\nu+1)} - v^*\|_{A,\eta} &\le (1 - \delta + \eta)\|v^{(\nu)} - v^*\|_{A,\eta} + O(\|v^{(\nu)} - v^*\|_{A,\eta}^2), \\ &\le (1 - \delta + \eta + O(\|v^{(\nu)} - v^*\|))\|v^{(\nu)} - v^*\|_{A,\eta}, \\ &\le (1 - \delta + \eta + O(\varepsilon))\|v^{(\nu)} - v^*\|_{A,\eta}. \end{aligned}$$

Here we have employed the equivalence of norms in finite-dimensional spaces. By choosing η and ε sufficiently small we can be assured that $1 - \delta + \eta + O(\varepsilon) < 1$, and thus our induction is completed. Further, we have shown that the iteration (3.8) is a strict contraction on $B_\varepsilon(u^*)$, and thus $v^{(\nu)} \to v^*$. □

4. Recursive estimation of the projectors *P* and *Q*. The stabilized iteration (2.14) requires the projectors P and Q onto the subspaces $\mathbb{P}$ and $\mathbb{Q}$. To obtain these we need only find an orthonormal basis for the subspace $\mathbb{P}$. Suppose that $Z \in \mathbb{R}^{N\times m}$ is an orthonormal basis for $\mathbb{P}$. Then the projectors P and Q are:

$$P = ZZ^T, \quad Q = I - ZZ^T \quad \text{with} \quad Z^TZ = I_m \in \mathbb{R}^{m\times m}. \tag{4.1}$$

In this section, we will show how an approximation to the basis Z may be computed and maintained recursively as continuation proceeds. An important feature of our technique is that we estimate Z directly from the iterates $q^{(\nu)}$ of the algorithm (2.14), without computing Jacobians.

4.1. Increasing the size of the basis. We assume that initially the continuation procedure starts with $\lambda = \lambda_a$ say, for which the fixed point procedure (2.3) or (2.14.2b) converges well. Then there is no unstable subspace $\mathbb{P}$, its dimension $m = 0$, and so $Z = 0$. As continuation in λ proceeds, we recursively determine a set of m, $m+1$, or $m+2$ basis vectors. This is done by monitoring the rate of convergence of the iterates $q^{(\nu)} \in \mathbb{Q}$ of (2.14.2b). If the rate degrades we know that some of the eigenvalues of $g_q^* = QF_u^*Q$ are approaching the unit circle. The generic case is that either an isolated real eigenvalue, μ_{m+1}, or a complex conjugate pair (μ_{m+1}, μ_{m+2}) approach $\{|z| = 1\}$. We must decide which is the case and determine the one or two vectors to adjoin to Z so that it now spans the larger invariant subspace of F_u^* associated with the augmented set of eigenvalues $\{\mu_1, \ldots, \mu_{m+1}\}$ or $\{\mu_1, \ldots, \mu_{m+1}, \mu_{m+2}\}$.

Let Y be an orthonormal basis for the invariant subspace associated with μ_{m+1} or the pair $\{\mu_{m+1}, \mu_{m+2}\}$. From (3.6) it easily follows that QY spans the invariant subspace of $g_q^* = QF_u^*Q$ associated with these eigenvalues, which are the dominant ones of g_q^*. Also, (Z, Y) and (Z, QY) span the same subspace, which is the desired augmented invariant subspace of F_u^*. An orthonormal basis for this subspace can therefore be obtained by adjoining to Z an orthonormal basis for the dominant invariant subspace of g_q^*. An approximation to such a basis is computed directly from the stabilized iteration (2.14) as described below.

Assume that for the current λ value $u^{(\nu)} = p^{(\nu)} + q^{(\nu)} \in B_\varepsilon(u^*(\lambda))$ and that $F_u(u, \lambda)$, and hence, g_p and g_q are Lipschitz continuous on $B_\varepsilon(u^*(\lambda))$. Then from (2.14.2b):

$$
\begin{aligned}
(4.2)\quad \triangle q^{(\nu)} &\equiv q^{(\nu+1)} - q^{(\nu)} \\
&= g(p^{(\nu)}, q^{(\nu)}, \lambda) - g(p^{(\nu-1)}, q^{(\nu-1)}, \lambda) \\
&= g(p^{(\nu-1)} + \triangle p^{(\nu-1)}, q^{(\nu-1)} + \triangle q^{(\nu-1)}, \lambda) - g(p^{(\nu-1)}, q^{(\nu-1)}, \lambda) \\
&= g_p^* \triangle p^{(\nu-1)} + g_q^* \triangle q^{(\nu-1)} + O(\varepsilon^2) \\
&= g_q^* \triangle q^{(\nu-1)} + O(\varepsilon^2).
\end{aligned}
$$

Here we have again used $g_p^* = QF_u^*P = 0$ from Lemma 2.10. Neglecting the $O(\varepsilon^2)$ terms in this gives on recursion

$$
(4.3)\qquad \triangle q^{(\nu)} \approx (g_q^*)^\nu \triangle q^{(0)}.
$$

Thus, the vectors $\{\triangle q^{(\nu)}\}$ are, to a second-order approximation in ε, a power iteration with the matrix $g_q^* = QF_u^*Q$ applied to the vector $\triangle q^{(0)}$. Therefore, asymptotically these vectors will tend to lie in the dominant eigenspace of g_q^*, provided the starting vector $\triangle q^{(0)}$ has a nonzero component in this direction.[1]

In practice, if the iteration fails to converge in say, $n_{\max}$ iterations, we use the two difference vectors $\{\triangle q^{(\nu)}, \triangle q^{(\nu-1)}\}$ (i.e., the two most recent power iterates) to approximate the dominant eigenspace of g_q^* as follows: we compute the Gram–Schmidt factorization (i.e., the "QR" factorization, computed by the "modified Gram–Schmidt" procedure [2]):

$$
(4.4)\qquad D \equiv [\triangle q^{(\nu)}, \triangle q^{(\nu-1)}] = \hat{D}T
$$

with $T \in \mathbb{R}^{2\times 2}$ upper triangular and $\hat{D} \in \mathbb{R}^{N\times 2}$ orthogonal. We examine the computed triangular matrix T, and if $T_{11} \gg T_{22}$ (in practice we used the test $|T_{11}| \geq 10^3 |T_{22}|$), we

[1] An example related to the exceptional case in which $\triangle q^{(0)}$ does not have a component in the dominant unstable subspace occurs in the second experiment in §8.

conclude that the dominant eigenspace of g_q* is one dimensional and we add *one* new vector, the first column of $\hat{D}$, to the basis Z. Otherwise, we conclude that the iteration is slowing down because a complex conjugate pair of eigenvalues is approaching the unit circle, and we add *two* new vectors to Z, the first two columns of $\hat{D}$. Note that in our current implementation we exclude the possibility that more than one real or two complex conjugate eigenvalues leave the unit disk simultaneously, but this could easily be altered.

The above procedure can possibly be made more accurate by using a better approximation to the dominant eigenvectors of g_q^* in span$\{\triangle q^{(\nu)}\}$ (which approximates a Krylov subspace for g_q^* [2]): to do this we accumulate a small window of r difference vectors $\{\triangle q^{(\nu)}\}_{\nu-r+1}^{\nu}$ while carrying out the iteration and compute an orthogonal basis U for span$\{\triangle q^{(\nu)}\}_{\nu-r+1}^{\nu}$ by the modified Gram–Schmidt procedure. We then compute $U^T F_u^* U$, which represents the restriction of F_u^* to span$\{\triangle q^{(k)}\}_{\nu-r+1}^{\nu}$. Next we determine the dominant eigenspace $\hat{U} \in \mathbb{R}^r$ or $\mathbb{R}^{r\times 2}$ of this $r \times r$ matrix. Finally, we form $U\hat{U}$ as our approximation and adjoin it to Z. However, this procedure involves extra evaluations of F in forming the matrix $U^T F_u^* U$. Since the original procedure above was found to produce a sufficiently accurate Z, we chose not to use this potentially more accurate but more expensive procedure.

Note that whenever the convergence rate of the iteration (2.14) degrades due to some large eigenvalues of g_q^*, the above procedure computes a basis for the corresponding invariant eigenspace. It does not, however, guarantee capturing the invariant subspace corresponding to exactly those eigenvalues outside the disk K_δ; in fact, δ plays no role in the above procedure. Nevertheless, approximating Z by this simple procedure accelerates the convergence of the iteration as desired, since the most unstable eigenmodes are either always captured or they are not yet present.

4.2. Maintaining accuracy of the basis. As the continuation progresses, the dominant eigenspace of F_u^* will change, rendering our estimate of the basis Z inaccurate. We correct for this by performing one step of an orthogonal power iteration [2] (also known as "*simultaneous*" or "*subspace*" iteration) on the columns of Z after each continuation step,

$$Z \leftarrow \text{orth}(F_u^* Z).$$

Here "orth$(F_u^* Z)$" denotes computing an orthonormal basis for the columns of $F_u^* Z$ using Gram–Schmidt orthogonalization. We will see in §5 that this step requires minimal extra work; in particular, no additional function evaluations are needed because an approximation to the quantity $F_u^* Z$ is required for executing the stabilized iteration. Further, we may easily monitor the accuracy of the eigenbasis by evaluating the matrix

$$\mathcal{E} \equiv Q F_u^* P,$$

which should vanish by Lemma 3.1 if $\mathcal{R}(Z)$ is in fact an invariant subspace of F_u^*. If $\mathcal{E}$ becomes too large, we can perform a few additional steps of the power iteration. However, this is expensive since it will require more function evaluations. In practice, we did not find it necessary; the one (free) power iteration per continuation step was sufficient to maintain Z to reasonable accuracy.

4.3. Decreasing the size of the basis. We must also consider decreasing the size of Z, since as we have pointed out in §2, it is important that the $\mathcal{R}(Z)$ be an invariant subspace of F_u^*, not merely contain an invariant subspace. After the dimension of Z

is increased due to eigenvalues of F_u^* approaching the unit circle, the algorithm will track the eigenspace via power iteration as long as these eigenvalues remain dominant. However, if some of these eigenvalues subsequently decrease in magnitude and rejoin the large cluster of small eigenvalues of F_u^*, it is likely that the power iteration will fail to separate the clustered eigenvalues. Consequently, the computed basis Z may then cease to span an invariant subspace, even though it might still contain the dominant eigenspace. Further, the procedure described in §4.1 can overestimate the dimension of the basis Z. We therefore test to decrease the size of Z by introducing the $m \times m$ matrix

$$H = Z^T F_u^* Z = Z^T f_p^* Z. \tag{4.5}$$

An approximation to H is computed by the numerical algorithm (see §5). The eigenvalues of H are a subset of those of F_u^*. We would like them to be only those outside the disk K_δ. After each continuation step we compute the eigenvalues and eigenvectors of H (this is inexpensive since H is a small $m \times m$ matrix). If only $\hat{m} < m$ eigenvalues of H lie outside K_δ, we compute a real basis $V \in \mathbb{R}^{m \times \hat{m}}$ for the corresponding eigenvectors of H (i.e., if the eigenvalues are complex, we compute a real basis rather than a basis of eigenvectors). Then span$\{ZV\}$ will be a good approximation to the desired eigenbasis, and we replace Z by

$$Z \leftarrow \text{orth}(ZV).$$

In this fashion, the size of the eigenbasis is reduced automatically when required.

5. The numerical algorithm: RPM. In this section, we describe the *Recursive Projection Method* or RPM, which is the numerical algorithm that implements the stabilization procedure using the projections computed recursively by the techniques of §4.

The algorithm of §4 actually computes $\hat{Z}$, an approximation to Z, and the corresponding projectors:

$$\hat{P} = \hat{Z}\hat{Z}^T \quad \text{and} \quad \hat{Q} = I - \hat{Z}\hat{Z}^T. \tag{5.1}$$

Thus, the numerical algorithm uses the orthogonal direct sum decomposition:

$$\mathbb{R}^N = \hat{P}\mathbb{R}^N \oplus \hat{Q}\mathbb{R}^N \equiv \hat{\mathbb{P}} \oplus \hat{\mathbb{Q}}, \tag{5.2}$$

instead of (2.7), where $\hat{\mathbb{P}} \equiv \hat{P}\mathbb{R}^N$ is an approximation to the subspace $\mathbb{P}$.

We find that in the stabilized iteration (2.14) it is sufficient to compute $(I - f_p)^{-1}$ only once per continuation step, rather than update it after each iteration. In other words, in our implementation, we actually perform the special Newton or chord iteration on the subspace $\hat{\mathbb{P}}$, rather than exact Newton on $\mathbb{P}$ as described in §2.

The numerical stabilized iteration follows.

Numerical Stabilized Iteration.

1. $p^{(0)} = \hat{P}u^{(0)}(\lambda), \qquad q^{(0)} = \hat{Q}u^{(0)}(\lambda);$
2. Do until convergence:

(5.3)

(a) $p^{(\nu+1)} = p^{(\nu)} + (I - \hat{f}_p^{(0)})^{-1}(\hat{f}(p^{(\nu)}, q^{(\nu)}, \lambda) - p^{(\nu)}),$

(b) $q^{(\nu+1)} = \hat{g}(p^{(\nu)}, q^{(\nu)}, \lambda);$

3. $u^*(\lambda) = p^{(\nu_{\text{Final}})} + q^{(\nu_{\text{Final}})} \equiv p^* + q^*.$

Here we have used

$$\hat{f}(p, q, \lambda) \equiv \hat{P}F(p + q, \lambda) \quad \text{and} \quad \hat{g}(p, q, \lambda) \equiv \hat{Q}F(p + q, \lambda).$$

Note that this iteration requires us to invert $(I - \hat{f}_p^{(0)}) : \hat{\mathbb{P}} \to \hat{\mathbb{P}}$, which is the restriction of $(I - F_u^{(0)})$ to the subspace $\hat{\mathbb{P}}$. We shall assume that this inverse exists. It is easily verified that

$$(I - \hat{f}_p)^{-1} = \hat{Z}(I - \hat{Z}^T F_u \hat{Z})^{-1} \hat{Z}^T. \tag{5.4}$$

In the actual computation, we introduce coordinate variables z for the representation of $p \in \hat{\mathbb{P}}$ in the basis $\hat{Z}$:

$$z \equiv \hat{Z}^T p = \hat{Z}^T u, \qquad z \in \mathbb{R}^m \text{ so } p = \hat{Z}z \text{ and } u = \hat{Z}z + q. \tag{5.5}$$

The iteration (5.3a) in the subspace $\hat{\mathbb{P}}$ can be written in these variables using (5.4) and $\hat{Z}^T \hat{Z} = I$:

$$z^{(\nu+1)} = z^{(\nu)} + (I - \hat{Z}^T F_u^{(0)} \hat{Z})^{-1} \left(\hat{Z}^T F(u^{(\nu)}, \lambda) - z^{(\nu)} \right). \tag{5.6}$$

If the only code available to us for the problem (2.3) is one to evaluate $F(u, \lambda)$, we can approximate $\hat{Z}^T F_u \hat{Z}$ by differencing. With $\hat{Z} = [\hat{Z}_1, \ldots, \hat{Z}_m] \in \mathbb{R}^{N \times m}$ and $\epsilon > 0$ we form

$$F_u \hat{Z}_i \approx \frac{1}{\epsilon} \left[F(u + \epsilon \hat{Z}_i, \lambda) - F(u, \lambda) \right] \quad \text{for } i = 1, \ldots, m. \tag{5.7}$$

In this way applying F_u to each column of $\hat{Z}$ requires only one additional function evaluation, since $F(u, \lambda)$ is required in the remainder of the algorithm.

The complete algorithm that performs continuation in λ using the recursive projection method is summarized below, and we will refer to it as the RPM *Continuation algorithm*:

(5.8)
RPM Continuation algorithm$(u_{-1}, \lambda_{-1}, u_0, \lambda_0, n_{max}, \delta, \delta\lambda, tol)$
$\hat{Z} = [\,]$;
while ()
  secant_step(u, λ); $\nu \leftarrow 0$; $F \leftarrow F(u, \lambda)$;
  evaluate_derivative$(F_u \hat{Z})$; $\hat{H} \leftarrow \hat{Z}^T [F_u \hat{Z}]$;
  while $(\|(u - F)\|_2 > tol)$
    $z \leftarrow \hat{Z}^T u$; $q \leftarrow u - \hat{Z}z$; $\zeta \leftarrow \hat{Z}^T F$;
    numerical_stabilized_iteration:
      $z \leftarrow z + (I - \hat{H})^{-1}(\zeta - z)$;
      $q \leftarrow F - \hat{Z}\zeta$;
    $u \leftarrow \hat{Z}z + q$; $F \leftarrow F(u, \lambda)$; $\nu \leftarrow \nu + 1$;
    if $(\nu > n_{max})$
      increase_basis_size$(\hat{Z})$; $\nu \leftarrow 0$;
      evaluate_derivative$(F_u \hat{Z})$;
    endif
  endwhile
  if $(\sigma_k(\hat{H}) \in \{|z| < 1 - \delta\}$ for some $k)$
    decrease_basis_size$(\hat{Z})$;
  endif
  power_iteration_step$(\hat{Z})$;
  $(u_{-1}, \lambda_{-1}, u_0, \lambda_0) \leftarrow (u_0, \lambda_0, u, \lambda)$;
endwhile.

The algorithm requires as input two points (u_{-1}, λ_{-1}) and (u_0, λ_0) on the solution branch Γ, as well as parameters $n_{\max}, \delta$, a step size $\delta\lambda$ and a tolerance *tol*. In the algorithm, the procedure *numerical-stabilized-iteration* in (5.3) has been implemented using the coordinate variable z according to (5.6), and the definition of $\hat{Q}$ to write $\hat{Q}u = u - \hat{Z}z$, etc. We have abbreviated the procedures described above in this section and in sections §§2 and 4 as follows: *evaluate-derivative* in (5.7), *secant-step* in (2.2), *increase-basis-size* in §4.1, *decrease-basis-size* in §4.3 and *power-iteration-step* in §4.2. The variables modified by each procedure are included as its arguments, e.g., *secant-step* (u, λ) indicates that u and λ are updated by the procedure. Also $\sigma_k(\hat{H})$ is an eigenvalue of $\hat{H}$. We note that in the implementation given above, the step size $\delta\lambda$ is kept constant as continuation progresses; it may be of benefit to vary this, and some procedures for doing so are discussed in [7]. Finally, note that the algorithm requires solving linear systems only with the $m \times m$ matrix $(I - \hat{H})$ and avoids computing and inverting Jacobians of the system F, which are $N \times N$. Also, evaluations of F are kept to a minimum, since F is assumed to be a "black-box" code that is expensive to evaluate.

6. Analysis of the numerical algorithm. Recall that the numerical stabilization procedure used by the RPM algorithm as described in the previous section involves two significant deviations from the theoretical description of the procedure as introduced in §2:

A. The computed orthonormal basis $\hat{Z}$ does not span the dominant invariant subspace of F_u^*, but is an approximation to such a basis.

B. The "chord" (or "special" Newton) method is employed on the subspace $\hat{\mathbb{P}}$ instead of Newton's method on $\mathbb{P}$.

We now reexamine the convergence of the stabilization algorithm in the presence of the above approximations. Since the subspace $\hat{\mathbb{P}}$ is not an invariant subspace of F_u^*, the property $QF_u^*P = 0$ proved in Lemma 3.1 for P and Q does not hold for $\hat{P}$ and $\hat{Q}$. However, since the basis $\hat{Z}$ is computed to approximate Z, the computed projector $\hat{P}$ approximates P. We will use

$$\mathcal{E} \equiv \hat{Q}F_u^*\hat{P} = \hat{g}_p^* \tag{6.1}$$

as a measure of how well $\hat{\mathbb{P}}$ approximates an invariant subspace of F_u^*. Note that we can multiply (6.1) by $\hat{P}$ from the right, by $\hat{Q}$ from the left and use $\hat{P}^2 = \hat{P}$ and $\hat{Q}^2 = \hat{Q}$ to replace Lemma 3.1 by the next lemma.

LEMMA 6.2. *The projections $\hat{P}$ and $\hat{Q}$ satisfy*:

$$\hat{Q}(F_u^* - \mathcal{E})\hat{P} = 0. \tag{6.3}$$

It therefore follows that for any $p \in \hat{\mathbb{P}}$,

$$(F_u^* - \mathcal{E})p \in \hat{\mathbb{P}},$$

hence, $\hat{\mathbb{P}}$ is an invariant subspace for the perturbed matrix $F_u^* - \mathcal{E}$.

Let $\{\hat{\mu}_k\}_1^N$ be the eigenvalues of $F_u^* - \mathcal{E}$. We claim that the following assumptions are reasonable:

(6.4)
(a) $|\hat{\mu}_1| \geq |\hat{\mu}_2| \geq \cdots \geq |\hat{\mu}_m| > |\hat{\mu}_{m+1}| \geq \cdots \geq |\hat{\mu}_N|$.

(b) The columns of the computed basis $\hat{Z}$ span the invariant subspace of $F_u^* - \mathcal{E}$ associated with $\{\hat{\mu}_k\}_1^m$, i.e., the dominant subspace.

The first assumption will hold provided $\|\mathcal{E}\|$ is sufficiently small. The second assumption

is reasonable since the basis $\hat{Z}$ is computed by a power iteration, which approximates a basis for the dominant invariant subspace of F_u^*.

Instead of Lemma 2.10 we shall use the following.

LEMMA 6.5. *If*

(a) $F_u^* - \mathcal{E}$ *has a complete set of eigenvectors* S,

(b) *the eigenvalues* $\{\mu_k\}_1^N$ *of* F_u^* *satisfy* $|\mu_m| > |\mu_{m+1}| + 2\mathcal{K}\|\mathcal{E}\|$, *where* $\mathcal{K} = \|S\|\|S^{-1}\|$,

(c) $\mathcal{K}\|\mathcal{E}\| < \delta$ *where* δ *satisfies* (2.5),

then

$$\rho\left(\hat{Q}(F_u^* - \mathcal{E})\hat{Q}\right) < 1 - \delta + \mathcal{K}\|\mathcal{E}\| < 1. \tag{6.6}$$

Proof. By the Bauer–Fike theorem [1], [2] we know that the eigenvalues of the perturbed matrix $F_u^* - \mathcal{E}$ differ from the eigenvalues of F_u^* by at most $\mathcal{K}\|\mathcal{E}\|$. Thus, by hypothesis (b) above the dominant eigenvalues $\{\hat{\mu_k}\}_1^m$ of $F_u^* - \mathcal{E}$ approximate the dominant eigenvalues $\{\mu_k\}_1^m$ of F_u^*, and for all the eigenvalues we have:

$$\begin{aligned} &\text{(a)} \quad \text{for each } k \in [1,m],\ |\hat{\mu}_k - \mu_i| \le \mathcal{K}\|\mathcal{E}\| \ \text{ for some } i \in [1,m] \\ &\text{(b)} \quad \text{for each } k \in [m+1,N],\ |\hat{\mu}_k - \mu_i| \le \mathcal{K}\|\mathcal{E}\| \ \text{ for some } i \in [m+1,N]. \end{aligned} \tag{6.7}$$

Therefore, since the eigenvalues $\{\mu_k\}_1^m$ lie outside the disk $K_\delta = \{|z| \le 1 - \delta\}$ and $\{\mu_k\}_{m+1}^N$ lie within K_δ, using (6.7b) we obtain

$$|\hat{\mu}_k| \le 1 - \delta + \mathcal{K}\|\mathcal{E}\| \quad \text{for} \quad k \in [m+1,N]. \tag{6.8}$$

Further, using (6.7a) and the hypothesis (b) of the Lemma,

$$|\hat{\mu}_k| > 1 - \delta + \mathcal{K}\|\mathcal{E}\| \quad \text{for} \quad k \in [1,m]. \tag{6.9}$$

Since, by our assumption (6.4b) above, $\hat{Z}$ is a basis for the invariant subspace of $F_u^* - \mathcal{E}$ associated with eigenvalues $\{\hat{\mu_k}\}_1^m$, the result (6.6) can now be proved in a manner similar to the proof of Lemma 2.10. □

As before, we can write the numerical stabilized iteration (5.3) as

$$\begin{aligned} &\text{(a)} \quad p^{(\nu+1)} = \hat{h}(p^{(\nu)}, q^{(\nu)}, \lambda), \\ &\text{(b)} \quad q^{(\nu+1)} = \hat{g}(p^{(\nu)}, q^{(\nu)}, \lambda), \end{aligned} \tag{6.10}$$

with

$$\hat{h}(p,q,\lambda) \equiv p + (I - \hat{f}_p^{(0)})^{-1}(\hat{f}(p,q,\lambda) - p). \tag{6.11}$$

Recall that we have assumed $(I - \hat{f}_p^{(0)})$ to be nonsingular. Now we can evaluate the Jacobian $\partial(\hat{h},\hat{g})/\partial(p,q)$ at the solution (p^*,q^*) to get, using $\hat{f}(p^*,q^*,\lambda) = p^*$:

$$\begin{aligned} \left.\frac{\partial(\hat{h},\hat{g})}{\partial(p,q)}\right|_{p^*,q^*} &= \begin{pmatrix} I + (I - \hat{f}_p^{(0)})^{-1}(\hat{f}_p^* - I) & (I - \hat{f}_p^{(0)})^{-1}\hat{f}_q^* \\ \hat{g}_p^* & \hat{g}_q^* \end{pmatrix} \\ &\equiv \begin{pmatrix} \mathcal{E}_1 & (I - \hat{f}_p^{(0)})^{-1}\hat{f}_q^* \\ \mathcal{E} & \hat{g}_q^* \end{pmatrix}. \end{aligned} \tag{6.12}$$

An estimate of the spectral radius of this Jacobian is given in the next lemma.

LEMMA 6.13. *Under the assumptions of Lemma* 6.5, *suppose* F_u *is Lipschitz continuous,* $(I - f_p^{(0)})$ *is nonsingular,* $\partial(\hat{h}, \hat{g})/\partial(p, q)|_{p^*,q^*}$ *has a complete set of eigenvectors* $\hat{S}$, *and*

$$\hat{\mathcal{K}}\|\mathcal{E}\| = \|\hat{S}\|\|\hat{S}^{-1}\|\|\mathcal{E}\| < \delta - \mathcal{K}\|\mathcal{E}\|.$$

Then the Jacobian of the numerical stabilized iteration evaluated at the solution satisfies

$$\rho\left(\left.\frac{\partial(\hat{h}, \hat{g})}{\partial(p, q)}\right|_{p^*,q^*}\right) \leq 1 - \delta + \mathcal{K}\|\mathcal{E}\| + \hat{\mathcal{K}}\|\mathcal{E}\| < 1 \tag{6.14}$$

for all initial values $u^{(0)}$ *lying in a ball* $B_{\hat{\epsilon}}(u^*) = \{u : \|u - u^*\| \leq \hat{\epsilon}\}$.

Proof. The structure of the Jacobian is given by (6.12). Using the Lipschitz continuity of F_u, it is easily shown that $\mathcal{E}_1 = O(\|u^{(0)} - u^*\|)$. Thus, we can ensure $\rho(\mathcal{E}_1) < 1 - \delta$ by choosing the radius of the ball $B_{\hat{\epsilon}}$ sufficiently small. Now

$$\hat{g}_q^* = \hat{Q}F_u^*\hat{Q} = \hat{Q}(F_u^* - \hat{Q}F_u^*\hat{P})\hat{Q} = \hat{Q}(F_u^* - \mathcal{E})\hat{Q}$$

using the definition (6.1) of $\mathcal{E}$ and $\hat{P}\hat{Q} = 0$. By Lemma 6.5 $\rho(\hat{Q}(F_u^* - \mathcal{E})\hat{Q}) < 1 - \delta + \mathcal{K}\|\mathcal{E}\|$. Using the Bauer–Fike theorem [1], [2] once more we arrive at the result (6.14). □

We can now easily prove the following convergence theorem in a manner similar to the proof of Theorem 3.13.

THEOREM 6.15. *For some* $\hat{\varepsilon}_1 = \hat{\varepsilon}_1(\lambda) > 0$ *let* $F(u, \lambda)$ *satisfy on the ball* $B_{\hat{\varepsilon}_1}(u^*) = \{u : \|u - u^*\| < \hat{\varepsilon}_1\}$

(a) $F(u, \lambda)$ *is twice differentiable,*

(b) $1 \notin \sigma\{\hat{f}_p^*\} \equiv \sigma\{\hat{P}F_u(u^*(\lambda), \lambda)\hat{P}\}$.

Then the numerical stabilized iteration (5.3) *converges for all initial values* $u^{(0)} \in B_{\hat{\varepsilon}}(u^*)$ *for some* $\hat{\varepsilon} < \hat{\varepsilon}_1$.

7. Pseudo-arclength RPM continuation. We have developed our stabilization algorithm RPM assuming $(I - F_u)$ is nonsingular in a neighborhood of the solution $u^*(\lambda)$. However, if the iteration (2.3) becomes unstable due to a real eigenvalue crossing the unit circle at the point $z = 1$, then there will exist a λ^* where $[I - F_u(u^*(\lambda^*), \lambda^*)]$ is singular. We refer to such points as singular points, and they lead to folds and bifurcations. The generic case is that of folds for which pseudo-arclength continuation [6] enables continuation to proceed beyond the singular point. In this section, we demonstrate how pseudo-arclength methods can be incorporated in the RPM continuation algorithm. Following Jarausch and Mackens [4] we need only implement the arclength procedure in the subspace $\mathbb{P}$.

To allow continuation past singular points we introduce a new parameter s, and hence, can augment the decomposed system (2.9) with a new scalar constraint, resulting in the reordered system:

$$\begin{aligned} &\text{(a)} \quad q = g(p, q, \lambda), \\ &\text{(b)} \quad p = f(p, q, \lambda), \\ &\text{(c)} \quad N(p, \lambda, s) = 0. \end{aligned} \tag{7.1}$$

Since the restriction of $(I - F_u^*)$ to the subspace $\mathbb{P}$ is singular and the restriction to $\mathbb{Q}$ is not, we perform pseudo-arclength continuation only on this small dimensional subspace. Therefore, the pseudo-arclength normalization $N(p, \lambda, s) = 0$ is chosen (see (7.4)) to be independent of q. Now the solution branch parameterized by s is $\Gamma(s) : \{u^*(s), \lambda^*(s)\}$ with $u^*(s) = p^*(s) + q^*(s)$ and $\lambda^*(s)$ satisfying (7.1).

The pseudo-arclength RPM uses a predictor-solver scheme as before: given two points on Γ, say $(u_{-1}, \lambda_{-1}) \equiv (u^*(s_{-1}), \lambda^*(s_{-1}))$ and $(u_0, \lambda_0) \equiv (u^*(s_0), \lambda^*(s_0))$, we predict a new solution by the secant method:

$$
\begin{aligned}
&\text{(a)} \qquad \lambda^{(0)}(s) = \lambda_0 + \frac{\delta s}{s_0 - s_{-1}}(\lambda_0 - \lambda_{-1}),\\
&\text{(b)} \qquad u^{(0)}(s) = u_0 + \frac{\delta s}{s_0 - s_{-1}}(u_0 - u_{-1}),\\
&\text{(c)} \qquad s = s_0 + \delta s,
\end{aligned}
\tag{7.2}
$$

where δs is the step size in the pseudo-arclength variable s. The solver step uses Newton's method on the variables $(p, \lambda) \in \mathbb{P} \times \mathbb{R}$ in (7.1b), (7.1c). For $q \in \mathbb{Q}$ we use the fixed point iteration in (7.1a), with a first order correction term $g_\lambda^{(\nu)} \triangle\lambda^{(\nu)}$. This correction term is introduced because λ changes during the iteration. (This term also plays a role in deriving (7.10) for the convergence analysis.) The resulting iteration is:

$$
\begin{aligned}
&\text{(a)} \qquad q^{(\nu+1)} = g(p^{(\nu)}, q^{(\nu)}, \lambda^{(\nu)}) - g_\lambda^{(\nu)} \triangle\lambda^{(\nu)},\\
&\text{(b)} \qquad \begin{pmatrix} I - f_p^{(\nu)} & -f_\lambda^{(\nu)} \\ N_p & N_\lambda \end{pmatrix} \begin{pmatrix} \triangle p^{(\nu)} \\ \triangle\lambda^{(\nu)} \end{pmatrix} = \begin{pmatrix} f(p^{(\nu)}, q^{(\nu)}, \lambda^{(\nu)}) - p^{(\nu)} \\ -N(p^{(\nu)}, \lambda^{(\nu)}, s) \end{pmatrix}.
\end{aligned}
\tag{7.3}
$$

The initial iterates $p^{(0)}$ and $q^{(0)}$ are obtained from (7.2b) by the decomposition $u^{(0)}(s) = p^{(0)} + q^{(0)}$. Further $\triangle p^{(\nu)} \equiv p^{(\nu+1)} - p^{(\nu)}$, $\triangle\lambda^{(\nu)} \equiv \lambda^{(\nu+1)} - \lambda^{(\nu)}$, $f_p^{(\nu)} \equiv f_p(p^{(\nu)}, q^{(\nu)}, \lambda^{(\nu)})$, $g_\lambda^{(\nu)} \equiv g_\lambda(p^{(\nu)}, q^{(\nu)}, \lambda^{(\nu)})$, etc. In the actual computation (7.3b) is written using the coordinate variables $z \in \mathbb{R}^m$, as done before in (5.6).

We use the pseudo-arclength normalization[2]

$$
N(p, \lambda, s) = \dot{p}^T(p - p_0) + \dot{\lambda}(\lambda - \lambda_0) - (s - s_0),
\tag{7.4}
$$

where $(\dot{u}, \dot{\lambda}) = (\dot{p} + \dot{q}, \dot{\lambda})$ is the unit tangent vector along the solution path $\Gamma(s)$. Note that instead of the tangent vector $\dot{u}$ (as used in standard pseudo-arclength methods [6]), we use $\dot{p}$, the projection of the tangent onto the subspace $\mathbb{P}$. In practice, the tangent vector $\dot{u}$ (and hence $\dot{p}$) is only approximately known and other pseudo-arclength conditions can be used in place of the above (see [6] and §8).

Using (7.4) in (7.3), the iteration scheme can be written as

$$
\begin{aligned}
&\text{(a)} \qquad q^{(\nu+1)} = g^{(\nu)/} - g_\lambda^{(\nu)} \triangle\lambda^{(\nu)},\\
&\text{(b)} \qquad \begin{pmatrix} p^{(\nu+1)} \\ \lambda^{(\nu+1)} \end{pmatrix} = \begin{pmatrix} p^{(\nu)} \\ \lambda^{(\nu)} \end{pmatrix} - \left[M^{(\nu)}\right]^{-1} \begin{pmatrix} p^{(\nu)} - f^{(\nu)} \\ N^{(\nu)} \end{pmatrix},
\end{aligned}
\tag{7.5}
$$

where

$$
\text{(c)} \qquad M \equiv \begin{pmatrix} I - f_p & -f_\lambda \\ \dot{p}^T & \dot{\lambda} \end{pmatrix}.
$$

[2]We actually use weights θ and $(1 - \theta)$, $0 < \theta < 1$ on the first two terms in (7.4), but they play no role in the convergence analysis, so we have dropped them here. In computations, however, they can be important.

The crucial point is that M can be nonsingular when $(I-f_p)$ is singular (under reasonable conditions), and this is what enables the pseudo-arclength procedure to continue past (singular) fold points. To demonstrate this, we first recall the conditions that hold at a simple fold, say (u^*, λ^*) on a solution path of $u = F(u, \lambda)$. They are:

$$\begin{aligned} &\text{(a)} \quad \dim \mathcal{N}(I - F_u^*) = 1, \\ &\text{(b)} \quad F_\lambda^* \notin \mathcal{R}(I - F_u^*). \end{aligned} \tag{7.6}$$

Now it is easy to show, as in the proof in [6], that M^*, the evaluation of M at the fold, is nonsingular if

$$\begin{aligned} &\text{(a)} \quad \dim \mathcal{N}(I - f_p^*) = 1, \\ &\text{(b)} \quad f_\lambda^* \notin \mathcal{R}(I - f_p^*). \end{aligned} \tag{7.7}$$

Although (7.7a) follows from (7.6a) and the projection $f_p^* = PF_u^*P$, since the spectrum of F_u^* and f_p^* coincide outside the disk K_δ, it is not always the case that (7.7b) follows from (7.6b). However, it does follow if F_u^* is normal; more precisely, we have the following.

LEMMA 7.8. *Let* (u^*, λ^*) *be a simple fold point of* $u = F(u, \lambda)$. *Then* (7.7b) *holds if* F_u^* *is normal.*

Proof. Since F_u^* is normal, it has an orthonormal set of eigenvectors, which span $\mathbb{R}^N$. Thus, both subspaces $\mathbb{P}$ and $\mathbb{Q}$ are invariant under F_u^*, and hence,

$$PF_u^*Q = 0 \quad \text{and} \quad QF_u^*P = 0. \tag{7.9}$$

Now suppose $f_\lambda^* \in \mathcal{R}(I - f_p^*)$. Then for any $\xi \neq 0$, there is a solution $v \in \mathbb{P}$ of

$$(I - f_p^*)v + \xi f_\lambda^* = 0.$$

But since $(I - g_q^*) : \mathbb{Q} \to \mathbb{Q}$ is nonsingular (by Lemma 2.10), there is a always a unique solution $w \in \mathbb{Q}$ of

$$(I - g_q^*)w + \xi g_\lambda^* = 0.$$

Using $Pw = 0$ and $Qv = 0$ we can combine the above two equations to get

$$[I - (PF_u^*P + QF_u^*Q)]\,(v + w) + \xi(P + Q)F_\lambda^* = 0.$$

Further, we have from (7.9):

$$[I - (P + Q)F_u^*(P + Q)]\,(v + w) + \xi(P + Q)F_\lambda^* = 0.$$

That is,

$$[I - F_u^*](v + w) + \xi F_\lambda^* = 0.$$

Since $\xi \neq 0$, this contradicts (7.6b) and our result follows. □

It also follows that M^* is nonsingular when $(I - f_p^*)$ is nonsingular and F_u^* is normal. The proof of this again employs (7.9) and proceeds as in [6].

Even when F_u^* is not normal, which is the generic case, a small perturbation in N can usually correct matters.

Convergence of the stabilized procedure using pseudo-arclength continuation can be studied by examining the Jacobian of the scheme (7.5) evaluated at the solution

$(u^*, \lambda^*) = (p^* + q^*, \lambda^*)$. Since $p^* = f(p^*, q^*, \lambda^*)$ and $N^* = 0$ at a solution, the $N+1 \times N+1$ Jacobian becomes, after a bit of calculation:

$$\frac{\partial(q^{(\nu+1)}, p^{(\nu+1)}, \lambda^{(\nu+1)})}{\partial(q^{(\nu)}, p^{(\nu)}, \lambda^{(\nu)})}\bigg|_{(u^*,\lambda^*)} = \begin{pmatrix} g_q^* - g_\lambda^* e_{N+1}^T (M^*)^{-1} \begin{pmatrix} f_q^* \\ 0 \end{pmatrix} & 0 \\ (M^*)^{-1} \begin{pmatrix} f_q^* \\ 0 \end{pmatrix} & 0 \end{pmatrix}. \tag{7.10}$$

Here, e_{N+1} is the $N+1$st unit vector in $\mathbb{R}^{N+1}$. Since $\sigma(g_q^*) \subset K_\delta$, we can easily write conditions to ensure that the perturbation of g_q^* in (7.10) maintains the contractivity of the Jacobian, and hence, the convergence of the scheme (7.5). We do not present these details (i.e., another application of the Bauer–Fike theorem) as they are of no practical value for our methods. Finally, note that when F_u^* is normal we have by (7.9) that $f_q^* = PF_u^*Q = 0$ and the spectrum of the Jacobian (7.10) lies in K_δ.

8. Numerical experiments. We present two numerical examples in which the pseudo-arclength RPM continuation algorithm was used to enable a "black box" time integration code to compute stable and unstable steady-state solution branches with folds and bifurcations. (This application of the RPM is examined in Appendix A.)

In both examples, the iteration F in (2.3) was constructed using second- and third-order explicit Runge–Kutta formulas for integration and step size control, respectively, with an error tolerance of 10^{-5}. The time interval $\triangle t = t_{\nu+1} - t_\nu$ was fixed to be 0.1, and the variable Runge–Kutta steps integrated the dynamic system (1.3) over this interval for each ν. The parameters used in the implementation were $\delta s = 0.15, n_{\max} = 13, \delta = 0.5$, and $tol = 10^{-4}$. For computing the pseudo-arclength condition $N(p, \lambda, s)$ defined by (7.4),[3] the tangents $\dot{p}$ and $\dot{\lambda}$ were replaced by the secant approximations

$$\dot{p} \approx \frac{(p - p_0)}{(s - s_0)} \quad \text{and} \quad \dot{\lambda} \approx \frac{(\lambda - \lambda_0)}{(s - s_0)}.$$

The derivatives $f_\lambda^{(\nu)}$ and $g_\lambda^{(\nu)}$ in (7.3) were replaced by their values at $\nu = 0$ (thus replacing Newton's method by the chord method) and computed using differences as described in (5.7), with $\epsilon = 10^{-3}$. Recall that in our examples F arises from a time integration code with error tolerance $\epsilon_0 \equiv 10^{-5}$, so we used $\epsilon = O(\sqrt{\epsilon_0})$.

For our first example, we considered the Bratu problem in one dimension:

$$G(u, \lambda) = u_{xx} + \lambda e^u = 0 \quad \text{with} \quad u(0) = u(1) = 0. \tag{8.1}$$

We discretized the operator on a spatial grid of 40 points to generate the system of ordinary differential equations $\partial u/\partial t = G(u, \lambda)$ used by the Runge–Kutta time integration code. Note that the Jacobian F_u of the resulting fixed-point iteration is not symmetric, even though G_u is. The solution diagram computed by the RPM algorithm is shown in Fig. 1. Notice first that the code was able to continue past the fold point and compute the unstable branch. The $*$ in the plot mark the points where the code increased or decreased the size of the eigenbasis and circles denote singular points detected by the algorithm, i.e., points where eigenvalues of $\hat{H} = \hat{Z}^T F_u^* \hat{Z}$ cross the unit circle. The shading of the line itself indicates the number of iterations required to converge at a particular point, solid lines indicating fastest convergence (i.e., points where the iteration took less than five iterations to converge) and dotted lines indicating slowest convergence (i.e., greater than nine iterations).

[3] In the implementation weights, θ and $(1 - \theta)$ were used on the first two terms of (7.4), with $\theta = 0.1$.

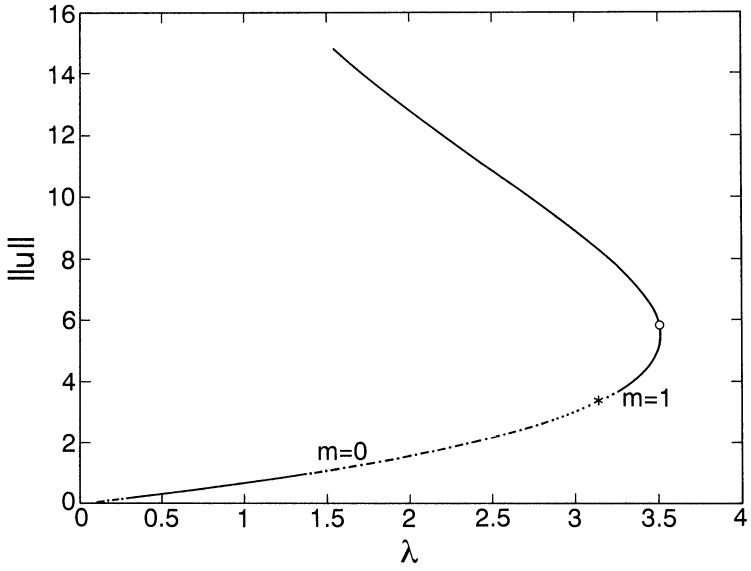

FIG. 1. *Problem* 1.

For our second example, we considered a more complicated system in which the Jacobian G_u is unsymmetric and some of the bifurcations present on the solution curve are Hopf bifurcations, where a pair of complex conjugate eigenvalues of G_u cross the imaginary axis. The system is the one-dimensional coupled set of equations in two variables u_1 and u_2:

$$
\begin{aligned}
G_1(u_1, u_2, \lambda) &= \tfrac{1}{5}(u_1)_{xx} + \lambda(u_1 - u_2) + \tfrac{1}{5}\lambda^2 e^{u_1} = 0, \\
G_2(u_1, u_2, \lambda) &= \tfrac{1}{5}(u_2)_{xx} + \lambda(u_1 + u_2) + \tfrac{1}{5}\lambda^2 e^{u_2} = 0,
\end{aligned} \tag{8.2}
$$

with boundary conditions $u_i(0) = u_i(1) = 0, i = 1, 2$. The operator was discretized on a spatial grid of 20 points to generate the coupled system of ordinary differential equations $\{\partial u_1/\partial t = G_1,\ \partial u_2/\partial t = G_2\}$ used by the time integration code. The computed solution diagram is shown in Fig. 2, with the same interpretation of symbols as in the previous example. (Note: the second circle from the left is superimposed on a $*$, indicating that the dimension of the basis was also changed at that point.)

As expected, for small values of λ the unsymmetric part of the system dominates, and the first two circles on the curve (from the left) denote Hopf bifurcation points detected by the algorithm, corresponding to two successive pairs of eigenvalues of F_u leaving the unit circle. (Note that the solution branch to the right of the first Hopf bifurcation is unstable, since the system has eigenvalues with positive real part.) The $\lambda^2 e^u$ terms begin to dominate as λ gets larger and because of this the system goes through a fold. Note the behavior of the algorithm for this example. The dimension of the eigenbasis $\hat{Z}$ is increased first to two, four, and then six, and subsequently decreased to five as an eigenvalue retreats within the disk of radius $1 - \delta$.

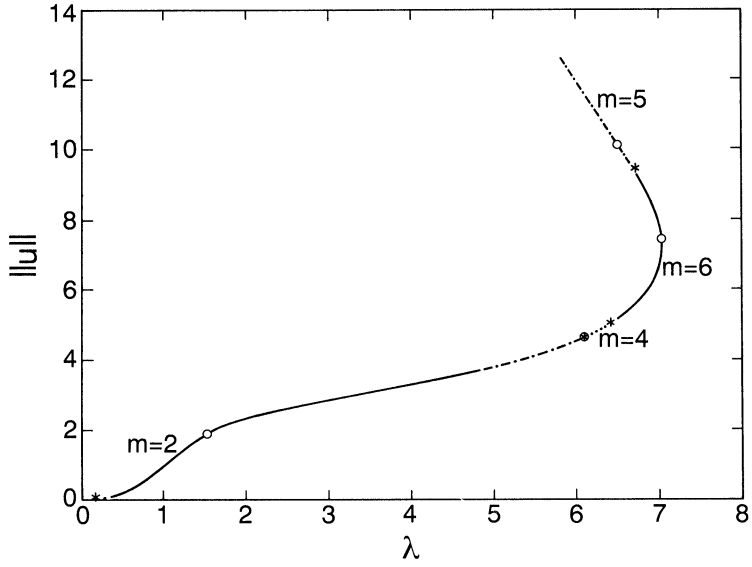

FIG. 2. *Problem 2.*

As a check we computed the solution diagrams for both problems using arclength continuation with a full Newton iteration on the steady-state equations $G(u, \lambda) = 0$, and confirmed the accuracy of the solution branches computed by the algorithm. For the second, nonsymmetric example, we also computed the eigenvalues of the Jacobian G_u at points on the solution path to verify the bifurcation points detected by the algorithm. We noted that the system $G(u, \lambda) = 0$ does in fact display the series of bifurcations shown in Fig. 2, two Hopf bifurcation points followed by a fold, and finally a transverse bifurcation point. Each Hopf bifurcation involves a pair complex conjugate eigenvalues leaving the unit disk; at the fold point a real eigenvalue leaves the unit disk. The final transverse bifurcation, on the other hand, occurs as a real eigenvalue re-enters the unit disk. The locations of the bifurcation points computed by the RPM algorithm were within one step size (i.e., 0.15) of the actual locations, except for the second Hopf bifurcation at $\lambda \approx 6.3$. This bifurcation actually occurs around $\lambda \approx 4.6$. Closer investigation reveals the reason for the large deviation and also provides deeper insight into the behavior of the algorithm.

We found that in the neighborhood of the second Hopf bifurcation, the solution u was *orthogonal* to the eigenspace corresponding to the pair of eigenvalues crossing the imaginary axis. Since we used a secant predictor step, the initial vectors for the numerical stabilized iteration had no component in this unstable subspace, and so the iteration (2.14) did *not* diverge at this bifurcation. Consequently, the algorithm found no reason to increase the dimension of the basis $\hat{Z}$ and did not "see" the bifurcation occur; the occurrence of the bifurcation was instead detected at a later point along the branch when the orthogonality of u to the relevant eigenspace ceased to hold, causing the iteration to

diverge. Although unusual, this behavior is understandable since the primary purpose of the algorithm is to compute solution branches efficiently, and bifurcation points are detected only through their effect on the convergence rate of the stabilized iteration. We have not yet attempted to incorporate sharper bifurcation detection tests into the RPM algorithm.

Finally, notice that in both examples the dimension of the eigenbasis $\hat{Z}$ was usually increased *before* the actual bifurcation point (except in the above case). Because of this, not only is the convergence past the bifurcation point assured, but the iteration is accelerated as soon as the algorithm detects that the convergence rate is degrading, before the bifurcation point itself is reached.

9. Discussion and conclusions. We have developed the RPM: a stabilization procedure that can extend the domain of convergence of fixed point iterations of the form (2.3). The RPM determines subspaces in which the iteration diverges and corrects for this by using Newton's method in the unstable subspace. The unstable basis vectors are computed efficiently from the iterates of the fixed point iteration. Further, the algorithm does not assume any special property of the iteration (2.3) or structure of the Jacobian F_u.

To enable continuation of solutions past singular points we have combined our RPM algorithm with pseudo-arclength continuation, which is used only on the unstable subspace.

As an application of our stabilization procedure we demonstrated the use of the algorithm in enabling a black box time integration code to compute solution diagrams of systems including folds, bifurcations, and unstable branches.

Certainly this approach has potential limitations. The algorithm is practical only when the number of divergent modes is small compared to the dimension of the system; nevertheless, this is the case in many applications.

Finally, we believe that the idea of stabilization of fixed-point iterations is more general. By identifying and isolating the slowly converging subspace we are greatly accelerating the iteration. So in a way we are adaptively constructing a quasi-Newton iteration, with just enough of Newton's method in it to give an acceptable convergence rate. Possible generalizations of this basic idea to the solution of stiff initial and boundary value problems, as well as in constructing adaptive preconditioners for solving sparse linear and nonlinear systems are subjects ripe for investigation.

Appendix A. Time integration to steady states. One standard way to find steady states of some dynamical phenomenon is to solve the appropriate initial value problem:

$$\text{(A.1)} \qquad \text{(a)}\ \frac{\partial u}{\partial t} = G(u,\lambda), \qquad \text{(b)}\ u(0,\lambda) = u^{(0)}(\lambda),$$

and to take the limit as $t \to \infty$ of the solution $u(t,\lambda)$. We assume here that the (in general) nonlinear partial differential operators on the right-hand side of (A.1a) have been approximated by means of an appropriate discretization procedure (i.e., finite differences, finite elements, spectral methods). Thus, we take $G : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N$, where $N \gg 1$ and λ is the parameter to be varied. The vector $u \in \mathbb{R}^N$ determines the spatial approximation to the original continuous quantities. So if finite differences with mesh h are employed and the original problem is over a spatial domain in $\mathbb{R}^d$, then $N \approx rh^{-d}$, where r is the number of continuous dependent variables. Note that N is not basically dependent on the nature of the continuous problem, but rather, on the accuracy with which we seek to determine the approximation. Of course if some sharp front develops

as λ varies we may wish to increase N, but this need not change its order of magnitude, and such phenomena do not play an important role in our study.

The local (linear) stability of a steady state, say $u^*(\lambda)$, of (A.1a) is governed by the eigenvalues $\{\alpha_k(\lambda)\}$ of the linearized eigenvalue problem:

$$G_u(u^*(\lambda), \lambda)\phi = \alpha\phi, \qquad \|\phi\| = 1. \tag{A.2}$$

Local stability occurs if and only if

$$\text{Re } \alpha_k(\lambda) < 0 \quad \text{for all } k \in [1, N],$$

and $u^*(\lambda)$ is unstable if

$$\text{Re } \alpha_k > 0 \quad \text{for some } k \in [1, N].$$

In general, the spectrum $\{\alpha_k(\lambda)\}$ varies continuously with λ, and at most, a small number of eigenvalues will cross the imaginary axis when λ changes by a small amount. Indeed, when a steady-state $u^*(\lambda)$ loses its stability as λ varies it usually occurs due to a single real or a complex conjugate pair of eigenvalues crossing the imaginary axis. Thus, we will assume that only m eigenvalues of (A.2) have positive real parts after $u^*(\lambda)$ has become unstable and $m \ll N$. Further, the variation of m with λ is essentially independent of the mesh for an accurate discretization of the original continuous problem.

The simplest time discretization of (A.1) is given by the forward Euler scheme with uniform time grid $t_n = n\triangle t$:

$$u^{n+1}(\lambda) = u^n(\lambda) + \triangle t G(u^n, \lambda). \tag{A.3}$$

Here, $u^n(\lambda) \equiv u^n$ is the approximation to $u(t_n, \lambda)$, and we note that this is in the form (2.3) with

$$F(u^n, \lambda) \equiv u^n + \triangle t G(u^n, \lambda). \tag{A.4}$$

We easily see that the numerical stability of (A.3) is equivalent to the convergence of the iteration (2.3) using (A.4), provided the initial value and the initial iterate $u^0(\lambda)$ are the same. Indeed, the linearized discretization about $u^*(\lambda)$ and the Jacobian $F_u(u^*, \lambda)$ have eigenvalues related by

$$\mu_k(\lambda) = 1 + \triangle t \alpha_k(\lambda) \quad \text{for } k = 1, 2, \ldots, N. \tag{A.5}$$

So if $\alpha_k = \rho_k + i\tau_k$ it follows that

$$\text{(A.6)} \qquad \text{(a) } |\mu_k(\lambda)| < 1 \text{ if and only if (b) } \rho_k(\lambda) < -\frac{\triangle t}{2}|\alpha_k(\lambda)|^2.$$

Thus, we find that (A.3) is stable for $u^*(\lambda)$, and (2.3) is converging in some neighborhood of the steady-state $u^*(\lambda)$ provided

$$\rho_k(\lambda) < 0 \quad \text{for } k = 1, 2, \ldots, N, \tag{A.7}$$

and the time step is restricted by:

$$\triangle t = \theta \min_{1 \le k \le N} 2\frac{|\rho_k(\lambda)|}{|\alpha_k(\lambda)|^2} \quad \text{for some } 0 < \theta < 1. \tag{A.8}$$

Note that precisely when the steady state $u^*(\lambda)$ of (A.1) loses its stability the numerical scheme (A.3) with initial value equal to the steady state, i.e., $u^0(\lambda) \equiv u^*(\lambda)$, cannot be made stable by any choice of $\triangle t$. We recall that such exchanges of stability occur at transcritical and supercritical bifurcations and in these cases usually only one real eigenvalue changes sign. Further, at a typical Hopf bifurcation the real part of a pair of complex conjugate eigenvalues changes sign. Thus, in general, m will change from zero to a small integer when the time integration procedure switches from stable to unstable as λ varies. For more accurate time integration (e.g., Runge–Kutta methods) the Courant condition as in (A.8) will be modified, but the stability loss still affects only a few eigenvalues. (See [8] for such results concerning hyperbolic systems of partial differential equations.) In §8 we have presented numerical experiments in which a realistic time integration procedure (using Runge–Kutta instead of Euler) is used to compute unstable steady states using the RPM continuation algorithm.

In practice it is often important to detect bifurcation points lying along the solution branch $\Gamma = \{u : G(u, \lambda) = 0\}$. Using the stabilization algorithm this can often be done efficiently as well. As mentioned above, most bifurcation points occur where one or more eigenvalues of $G_u^* = G_u(u^*(\lambda), \lambda)$ cross the imaginary axis. Recall that the eigenvalues of G_u^* and F_u^* are related by (A.5). Further, the eigenvalues of F_u^* outside the circle $\{|z| = 1 - \delta\}$ are precisely the eigenvalues of the computed $m \times m$ matrix $H = Z^T F_u^* Z$ of (4.5).

REFERENCES

[1] F. L. BAUER AND C. T. FIKE, *Norms and exclusion theorems*, Numer. Math., 2, 1960, pp. 137–141.

[2] G. GOLUB AND C. VAN LOAN, *Matrix Computations*, Johns Hopkins University Press, Baltimore, MD, 2nd ed., 1989.

[3] H. JARAUSCH AND W. MACKENS, *Numerical treatment of bifurcation problems by adaptive condensation*, in T. Küpper, H. D. Mittelmann, and H. Weber, H. eds., Numer. Methods for Bifurcation Problems, Birkhäuser Basel; 1984.

[4] ———, *Computing bifurcation diagrams for large nonlinear variational problems*, in P. Deuflhard and B. Engquist, eds., Large Scale Scientific Computing, Vol. 7 of Progress in Scientific Computing, Birkhäuser, Boston-Basel-Stuttgart, 1987.

[5] ———, *Solving large nonlinear systems of equations by an adaptive condensation process*, Numer. Math., 50(6) (1987), pp. 633–653.

[6] H. B. KELLER, *Numerical solution of bifurcation and nonlinear eigenvalue problems*, in P. Rabinowitz, ed., Applications of Bifurcation Theory, Academic Press, New York, 1977.

[7] ———, *Lectures on Numerical Methods in Bifurcation Problems*, published for the Tata Institute of Fundamental Research, Bombay, by Springer-Verlag, New York, 1987.

[8] H-O. KREISS AND G. SCHERER, *Method of lines for hyperbolic differential equations*, SIAM J. Numer. Anal., 39(3) (1992), pp. 640–646.